

R Notebook

Gabriela Suardi

2023-03-08

In this file I'll put my notes from John Hopkins' data science course R programming

R console Input and evaluation

<- assignment operator : assign a value to a symbol “#” indicates a comment. Everything on the right of this symbol is ignored Example:

```
x<- 1      ## nothing printed
x          ## autoprinting occurs
```

```
## [1] 1
```

```
print("The value of x is")      ##explicited printing
```

```
## [1] "The value of x is"
```

```
print(x)
```

```
## [1] 1
```

The : operator is used to create interger sequences Example:

```
x <- 1:20  #this gives us a sequence
x
```

```
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

###R objects and attributes

R has five basic classes of objects: **-character -numeric (real numbers) -integer - complex -logical (true/false)**

The most basic object is a *vector* - Everything in R is a object -A vector can only contain objects of the *same* class. Example: a vector containing characters and numerics.

BUT the only exception is a **list**, which is represented as a vector but can contain objects of different classes Empty vectors can be created with the *vector()* function

The vector function has two basic arguments. The **first argument** is the *class* of the object, so the type of object that you want to have in the vector. And the **second argument** is the *length* of the vector itself.

####Numbers

Numbers in R are generally treated as *numeric objects*. If you explicitly want an integer, you need to use the suffix **L**. Ex: 1L. There's also the special number **inf** which represents infinity. The **NaN** (not a number) represents an undefined value or a missing value.

###Creating vectors

The `c()` function (concatenating) can be used to create vectors of an object

```
a <- c(0.5, 0.6) ##number
a
```

```
## [1] 0.5 0.6
```

```
b <- c(TRUE, FALSE) ##logical
d <- c(T, F)      ##logical
e <- c("1", "a", "d", "0.8") ##character
f <- 9:29 ##integer
f
```

```
## [1] 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
```

```
g <- c(1+0i, 2+4i) ##complex
g
```

```
## [1] 1+0i 2+4i
```

Using the vector function:

```
x <- vector("numeric", length=10)
x
```

```
## [1] 0 0 0 0 0 0 0 0 0 0
```

When different objects are mixed in a vector, **coercion** occurs so every object is of the same class. When converted to numeric, **TRUE** is represented by the number 1 and **FALSE** by 0.